Journal of Computer and System Sciences **56**, 272–276 (1998)
Article No. SS981568

Arun Sharma

School of Computer Science and Engineering, The University of New South Wales, Sydney, New South Wales 2052, Australia
E-mail: arun@cse.unsw.edu.au

Received September 5, 1996; revised December 19, 1997

According to Gold's criterion of identification in the limit, a learner, presented with data about a concept, is allowed to make a finite number of incorrect hypotheses before converging to a correct hypothesis. If, on the other hand, the learner is allowed to make only one conjecture which has to be correct, the resulting criterion of success is known as finite identification

Identification in the limit may be viewed as an idealized model for incremental learning whereas finite identification may be viewed as an idealized model for batch learning. The present paper establishes a surprising fact that the collections of recursively enumerable languages that can be finite identified (batch learned in the ideal case) from both positive and negative data can also be identified in the limit (incrementally learned in the ideal case) from only positive data.

It is often difficult to extract insights about practical learning systems from abstract theorems in inductive inference. However, this result may be seen as carrying a moral for the design of learning systems, as it yields, in the *ideal* case of no inaccuracies, an algorithm for converting batch systems that learn from both positive and negative data into incremental systems that learn from only positive data without any loss in learning power. This is achieved by the incremental system simulating the batch system in incremental fashion and using the heuristic of ''localized closed-world assumption'' to generate negative data.

Key words: inductive inference; computational learning theory; identification in the limit; finite identification; incremental learning; batch learning.

1. INTRODUCTION

Consider a scenario in which a machine **M** is learning a concept C. At any given time, a finite piece of data about C is made available to **M**. **M** reacts to these data by conjecturing a hypothesis about C. Availability of additional data may cause **M** to revise its currently held hypothesis. **M** is said to learn C just in case the sequence of hypotheses conjectured by **M** eventually stabilizes to a final hypothesis which is a correct representation of C. This is essentially Gold's [Gol67] seminal criterion of *identification in the limit*, and it may be viewed as an idealized model of incremental learnability.[1]

[1] This is an idealized model because the learner has access to all the data seen so far in revising its conjectures. More pragmatic models of incremental learnability restrict the number of previous data elements that a learner has access to in revising its current hypothesis. Such models are referred to as *memory-limited learning* and *iterative learning* in the literature (e.g., see [CJLZ97]).

It should be noted that success according to the identification in the limit criterion does not require the learner to confirm or signal its convergence. Such an additional constraint is equivalent to requiring the learner to output only one hypothesis which is correct. This latter criterion of success, referred to as *finite identification*, was first defined by Gold [Gol67] and investigated by Trakhtenbrot and Barzdin [TB70]. According to this criterion, a learner, presented with data about a concept in any order, is required to conjecture a single hypothesis that is a correct representation of the concept. The learner may wait long enough to issue its first hypothesis, but once it has committed to a hypothesis, no mind change is allowed. Systems that learn according to the criterion of finite identification are also referred to as "one-shot learners." The requirement that a learner's first conjecture be correct makes finite identification an idealized model of batch learning.

The result reported in the present paper shows that the collections of recursively enumerable languages that can be finite identified from both positive and negative data can also be identified in the limit from only positive data. We establish this result by giving an algorithmic transformation of a machine $\mathbf{M}_1$ into machine $\mathbf{M}_2$ such that $\mathbf{M}_2$ identifies in the limit from positive data every language that $\mathbf{M}_1$ finite identifies from both positive and negative data. This transformation yields an algorithm to convert a batch learning system that learns from both positive and negative data into an incremental learning system that learns from only positive data. Although we present our result in the general setting of learning grammars (accepting procedures) for recursively enumerable languages, it holds for every countable domain.

We now proceed formally. We introduce the preliminary notions of identification in the limit, finite identification, positive data, positive and negative data in Section 2. The result is presented in Section 3, followed by a discussion in Section 4.

2. PRELIMINARIES

Readers familiar with the inductive inference literature [AS83, OSW86] may wish to skip this section.

Let N denote the set of natural numbers, $\{0, 1, 2, \ldots\}$. As already noted, our domain is the collection of recursively

0022-0000/98 $25.00

enumerable (r.e.) languages over N. A grammar for an r.e. language L is a computer program that accepts L (or, equivalently, generates L [HU79]). For any r.e. language L, the elements of L constitute its positive data and the elements of the complement, $N-L$, constitute its negative data. We next formalize the notions of positive data presentation, positive and negative data presentation, learning machine, finite identification, and identification in the limit.

The totality of all positive data about a language may be conceived as a listing of its elements in some arbitrary order. To allow such lists to contain pauses in the presentation of data, we adjoin a nonnumeric element $\#$ to N. Such listings of positive data are captured in the notion of texts formally defined below.

DEFINITION 1. (a) A *text* is any mapping from N into $(N \cup \{\#\})$. A text may be visualized as an infinite sequence of members of $N \cup \{\#\}$. The typical variable for texts is T.

(b) The *content* of a text T, denoted content (T), is the set of natural numbers in the range of T.

(c) Let $L \subseteq N$ and a text T be given. We say that T is *for* L just in case content$(T) = L$.

Texts are infinite objects.[2] A learning machine, at any given time, can only be presented with finite amount of data. Hence, it is useful to have the notion of finite initial sequence of a text. This is the subject of the next definition.

DEFINITION 2. (a) Let text T and $n \in N$ be given. The initial finite sequence of length n is denoted $T[n]$.

(b) The set $\{T[n] \mid T \text{ is a text and } n \in N\}$ is denoted SEQ. We let σ and τ range over SEQ. Λ denotes the empty sequence.

(c) The *length* of σ, denoted $|\sigma|$, is the number of elements in σ.

(d) The *content* of a sequence σ, denoted content(σ), is the set of natural numbers in the range of σ.

(e) For $n < |\sigma|$, the initial sequence of length n in σ is denoted $\sigma[n]$.

The result of concatenating τ onto the end of σ is denoted $\sigma \diamond \tau$. We say that $\sigma \sqsubseteq \tau$ just in case σ is an initial sequence of τ, that is, $|\sigma| \leqslant |\tau|$ and $\sigma = \tau[|\sigma|]$.

Now, SEQ is the collection of all finite sequences over $(N \cup \{\#\})$. Since this collection is countably infinite, there exists a bijection between SEQ and N. Such a bijection assigns a canonical index to each member of SEQ. Henceforth, we will abuse the notation somewhat and refer to a sequence by its canonical index. We next describe what we mean by a learning machine.

[2] The only text for the empty language is $\#, \#, \#, \#, \ldots$. In fact the idea of adjoining $\#$ to N was first introduced by Blum and Blum [BB75] with a view to assign a text to the empty language.

A learning machine may be viewed as an algorithmic device that, when presented with a finite sequence of data about a concept, may issue a hypothesis about the concept from which the data is drawn. Since the concepts that these machines learn are r.e. languages, we take computer programs that accept languages as suitable hypotheses. Now, if we fix the programming language in which these hypotheses are expressed, then there is a bijection between the set of all computer programs and N. That is, we can think of each program being "named" by a number. In the case that the machine does not issue any hypothesis, we say that the machine outputs a special nonnumeric symbol, $\perp$. Thus, learning machines may be viewed as algorithmic devices that take numbers (members of SEQ) as input and output either numbers or $\perp$. This is summarized in the following definition.

DEFINITION 3. A *learning machine* is an algorithmic device that computes a mapping from SEQ into $N \cup \{\perp\}$.

We now turn our attention to the various criteria for a learning machine to be successful on a language. We first introduce the criterion of finite identification from texts in stages (Definitions 4 and 5).

DEFINITION 4 [Gol67]. A learning machine $\mathbf{M}$ is said to **TxtFin**-*identify* a language L just in case for each text T for L, there exists n_0 such that the following hold:

(a) for all $n < n_0$, $\mathbf{M}(T[n]) = \perp$;

(b) $\mathbf{M}(T[n_0])$ is a grammar for L; and

(c) for all $n \geqslant n_0$, $\mathbf{M}(T[n]) = \mathbf{M}(T[n_0])$.

We write $L \in \mathbf{TxTFin}(\mathbf{M})$.

In the notation **TxtFin**, **Txt** denotes positive data and **Fin** denotes the criterion of finite identification. So, a machine $\mathbf{M}$ **TxtFin**-identifies a language L just in case $\mathbf{M}$, fed any text for L, waits long enough to see sufficient amount of data before making its first conjecture which is a correct grammar for L. The reader should note that in the above definition once the machine has issued its first conjecture, we require it to continue with the same conjecture. This is equivalent to requiring the machine to output $\perp$ forever after it has issued its only conjecture.

The reader should also note that the above definition deals with the learnability of a single language. It is easy to see that every language L is learnable in the above sense; a constant machine that in response to any input outputs nothing but a grammar for L learns L. Clearly, such machines are not very useful, as they learn only one language and nothing else. To use an analogy from Osherson *et al.* [OSW86], these machines are like the village idiot who predicts an earthquake every day and happens to be correct on the day of the earthquake. Clearly, we require a notion that describes learnability of a collection of languages. The following definition achieves precisely this.

DEFINITION 5. (a) A learning machine **M** **TxtFin**-*identifies* a collection of languages $\mathscr{L}$ just in case **M** **TxtFin**-identifies each language in $\mathscr{L}$.

(b) **TxtFin** is the class of sets of languages $\mathscr{L}$ such that there exists a machine **M** that **TxtFin**-identifies $\mathscr{L}$. In other words, $\mathbf{TxtFin} = \{\mathscr{L} \mid (\exists \mathbf{M})[\mathscr{L} \subseteq \mathbf{TxtFin}(\mathbf{M})]\}$.

The class **TxtFin** above is the set-theoretic summary of the collections of languages that can be **TxtFin**-identified. Intuitively, if some collection of languages $\mathscr{L} \in \mathbf{TxtFin}$, then there exists a machine that **TxtFin**-identifies each language in $\mathscr{L}$.

We next introduce the criterion of identification in the limit from positive data.

DEFINITION 6 [Gol67]. (a) A machine **M** is said to **TxtEx**-*identify* a language L just in case for each text T for L, there exists a grammar G for L, such that for all but finitely many n, $\mathbf{M}(T[n]) = G$. We say that $L \in \mathbf{TxtEx}(\mathbf{M})$.

(b) A machine **M** **TxtEx**-*identifies* a collection of languages $\mathscr{L}$ just in case **M** **TxtEx**-identifies each language in $\mathscr{L}$.

(c) **TxtEx** is the class of sets of languages $\mathscr{L}$ such that there exists a machine **M** that **TxtEx**-identifies $\mathscr{L}$.

In **TxtEx**, **Txt** denotes positive data and **Ex** denotes identification in the limit. We next turn our attention to learning from both positive and negative data.

Texts represent an abstraction of the totality of positive data about a language. Similarly, the totality of both positive and negative data can be represented by *informants*. The notion of informant, defined below, was first considered by Gold [Gol67]. Like texts, informants are infinite sequences, but consist of #'s and ordered pairs of the form (n, x), where either $x = 1$ (denoting n is a positive data) or $x = 0$ (denoting n is a negative data). More precisely:

DEFINITION 7. (a) An informant I is an infinite sequence of #'s and ordered pairs (repetitions allowed) such that for each $n \in N$, either $(n, 0)$ or $(n, 1)$, but not both, appears in I.

(b) If I is an informant then content$^+(I)$ denotes the set of all such numbers n such that ordered pair $(n, 1)$ appears in I and content$^-(I)$ denotes the set of all numbers m such that ordered pair $(m, 0)$ appears in I.

(c) An informant I is for a language L just in case content$^+(I) = L$. Note that if I is for L then content$^-(I) = N - L$.

It is also useful to introduce finite initial sequences of informants. To this end, we adapt the machinery introduced for texts to informants and define

$$\mathrm{INFSEQ} = \{I[n] \mid I \text{ is an informant and } n \in N\}.$$

Hence, learning machines that learn from both positive and negative data compute algorithmic mappings from INFSEQ into $N \cup \{\perp\}$. We let γ and δ range over INFSEQ. If $\gamma \in$ INFSEQ, then content$^+(\gamma)$ denotes the set of numbers n such that $(n, 1)$ is in the sequence γ and content$^-(\gamma)$ denotes the set of numbers m such that $(m, 0)$ is in the sequence γ. The following definition describes the criterion of finite identification from both positive and negative data.

DEFINITION 8. (a) A learning machine **M** is said to **InfFin**-*identify* a language L just in case for each informant I for L, there exists n_0 such that the following hold:

(i) for all $n < n_0$, $\mathbf{M}(I[n]) = \perp$;

(ii) $\mathbf{M}(I[n_0])$ is a grammar for L; and

(iii) for all $n \geqslant n_0$, $\mathbf{M}(I[n]) = \mathbf{M}(I[n_0])$.

We write $L \in \mathbf{InfFin}(\mathbf{M})$.

(b) A learning machine **M** **InfFin**-*identifies* a collection of languages $\mathscr{L}$ just in case **M** **InfFin**-identifies each language in $\mathscr{L}$.

(c) **InfFin** is the class of sets of languages $\mathscr{L}$ such that there exists a machine **M** that **InfFin**-identifies $\mathscr{L}$.

In **InfFin**,**Inf** denotes both positive and negative data and **Fin** denotes the criterion of finite identification. Similarly, one can define **InfEx**-identification modeling idealized incremental learning from both positive and negative data.

3. RESULT

To summarize we have introduced the following notions:

(a) batch from texts (**TxtFin**-identification);

(b) batch from informants (**InfFin**-identification);

(c) incremental from texts (**TxtEx**-identification); and

(d) incremental from informants (**InfEx**-identification).

The subject of the present paper is the relationship between the classes **InfFin** and **TxtEx**. Relationships between other classes were known and are part of the folklore in the inductive inference literature (following from results in [Gol67] and results about function identification by Case and Smith [CS83]; see also [OSW86]). They are summarized below.

$$\begin{array}{ccc} \mathbf{TxtFin} & \subset & \mathbf{InfFin} \\ \cap & & \cap \\ \mathbf{TxtEx} & \subset & \mathbf{InfEx} \end{array}$$

From the above, the only relationship that is not clear is between **InfFin** (batch learning from both positive and negative data) and **TxtEx** (incremental learning from only positive data). We show that **InfFin** is properly contained in **TxtEx**.

THEOREM 1. $\mathbf{InfFin} \subset \mathbf{TxtEx}$.

We first give an informal sketch of the idea behind proof of **InfFin** $\subseteq$ **TxtEx** followed by a formal presentation. To this end, for each r.e. language, it is useful to point out a special informant, called the canonical informant. The canonical informant for a language L is such that for each $n \in N$, the nth ordered pair describes the status of n in L; that is, if $n \in L$ then the nth ordered pair is $(n, 1)$, otherwise it is $(n, 0)$. For example, the canonical informant for E, the set of even numbers, is

$$(0, 1), (1, 0), (2, 1), (3, 0), (4, 1), (5, 0), (6, 1), \ldots.$$

Let us suppose that a learning machine **M** is given that batch learns a collection of languages from informants. We then describe a machine **M**′ that incrementally learns the same collection from texts.

Since **M** batch learns L from informants, **M** is successful on every informant for L. In particular, **M** is successful on the canonical informant for L. Hence, there is an initial fragment of the canonical informant for L that causes the batch machine to issue its only correct hypothesis. Let us denote this initial fragment by γ. Now, the incremental machine **M**′ on initial segments of a given text for L performs the following two tasks:

- constructs a candidate for γ from the initial portion of the text seen so far;
- feeds the candidate for γ constructed above to the batch machine **M** and if **M** outputs a hypothesis then **M**′ outputs the same hypothesis.

It is clear that if **M**′ succeeds in constructing γ, it will succeed in learning the language. **M**′ attempts to construct γ by employing the heuristic of "localized closed-world assumption." This heuristic is best illustrated with the help of an example. Suppose the finite sequence

$$2, 6, 4$$

is an initial segment of a text for L. This means that elements 2, 4, and 6 are positive data for L. The heuristic of localized closed-world assumption says that everything not explicitly tagged as positive data up to the greater of

- the maximum element in the sequence, and
- the length of the sequence

is negative data. Hence, the use of this heuristic yields the following candidate for γ:

$$(0, 0), (1, 0), (2, 1), (3, 0), (4, 1), (5, 0), (6, 1).$$

Suppose the next element in the sequence is 0, that is, the initial sequence of the text for L is

$$2, 6, 4, 0.$$

M′ then applies the heuristic of localized closed-world assumption on this sequence to obtain the following candidate for γ:

$$(0, 1), (1, 0), (2, 1), (3, 0), (4, 1), (5, 0), (6, 1).$$

In this way **M**′ keeps on revising its candidate for γ as it sees more and more positive data. It is not too difficult to see that after examining sufficient portion of any text for L, **M**′ will hit upon the right candidate for γ—the initial segment for the canonical informant for L that caused the batch machine **M** to make its correct conjecture. We now proceed formally.

Proof. It is well known that **TxtEx** $-$ **InfFin** $\neq \emptyset$. Consider **FIN**, the class of finite languages. It is easy to show that **FIN** $\in$ **TxtEx** and **FIN** $\notin$ **InfFin**. So we only show that **InfFin** $\subseteq$ **TxtEx**.

Let the collection of languages $\mathscr{L} \in$ **InfFin**. Let machine **M** **InfFin**-identify $\mathscr{L}$. We construct a machine **M**′ that **TxtEx**-identifies $\mathscr{L}$.

The behavior of machine **M**′ discussed above is formally described below. The reader should note that in the following construction we employ a somewhat conservative version of the localized closed-world assumption because this conservative version is sufficient for our purpose. According to this version, given an initial sequence of a text, σ, we construct a candidate initial sequence for the canonical informant of length $|\sigma| + 1$ instead of length $\max(\max(\text{content}(\sigma)), |\sigma|) + 1$ as described above. This conservative version leads to fewer space requirements.

Begin M′(σ)
  M′ constructs γ, a candidate initial segment of the canonical informant as follows:
  Initialize $\gamma := \Lambda$.
  for $x := 0$ **to** $|\sigma|$ **do**
  if $x \in \text{content}(\sigma)$ **then**
    $\gamma := \gamma \diamond (x, 1)$
  else
    $\gamma := \gamma \diamond (x, 0)$.
  Search for the least length γ' such that $\gamma' \sqsubseteq \gamma$ and **M**$(\gamma') \neq \perp$.
  If search for γ' is successful **then**
    Output **M**(γ')
  else
    Output $\perp$.
End M′(σ)

Let $L \in \mathscr{L}$. Now, since **M** **InfFin**-identifies $\mathscr{L}$, **M** **InfFin**-identifies L on each informant for L. This implies that **M** **InfFin**-identifies L on its canonical informant. Let γ_0 be the

initial sequence of the canonical informant of L that causes **M** to output its (only) correct conjecture on the canonical informant for L. Let T be any text for L. Let n_0 be such that $n_0 > |\gamma_0| \wedge \text{content}^+(\gamma_0) \subseteq \text{content}(T[n_0])$. It is easy to verify that for all $n \geqslant n_0$, $\mathbf{M}'(T[n])$ is the same as the (only) conjecture emitted by **M** on the canonical informant for L. Clearly, $\mathbf{M}'$ **TxtEx**-identifies L. ▌

4. DISCUSSION

The surprising nature of the above result is evident from the fact that if the batch learning system is allowed to change its mind just once, the result does not hold. To see this, consider the collection of languages **COSINGLE** $\cup \{N\}$, where **COSINGLE** is the collection of all cosingleton languages.[3] Using a locking sequence argument, it can be shown that this collection of languages cannot be identified in the limit from only positive data, that is, **COSINGLE** $\cup \{N\} \notin$ **TxtEx** (see [OSW86] for a proof). However, it is easy to check that a batch learning system that has the option of changing its mind once can learn this collection from both positive and negative data.

Another interesting point about the result is that in the *ideal* case it yields an algorithm for transforming a batch learning system that learns from informants into an incremental learning system that learns from texts. Although the result was established for the task of learning grammars for r.e. languages, it is easy to see that it is applicable to other learning domains. For example, in the context of learning Horn clause axiomatizations for h-easy models from ground positive and negative facts (see Shapiro [Sha81]), a batch learning system can be converted into an incremental system that learns from only positive facts. The only point to note here is that a canonical informant for every h-easy model exists since the corresponding set of all ground facts for the underlying language is countable and therefore a bijection between the set of all ground facts and N can be defined.

As the result stands, it applies only to the ideal case of batch systems that learn from noise free data. In many practical applications, generating negative examples is a very tedious process. For this reason, applications of practical batch learning systems like FOIL [Qui90] and GOLEM [MF90], which are designed to operate from both positive and negative data, usually resort to the closed-world assumption to generate negative data. An interesting direction of research is to determine whether an adaptation of the result presented here to more realistic learning models incorporating inaccuracies in data sheds any light on this practice of using the closed-world assumption to generate negative data. More specifically, it would be interesting to find what can be said about the accuracy of incremental systems that evolve from batch systems by application of the closed-world assumption in the presence of inaccuracies in data. The only moral that can be drawn from the present result is that in noise-free environments, using the closed-world assumption is "acceptable" provided the system is run "several" times with increasing amounts of positive data.

Finally, we direct the reader to a related result by Lange and Zeugmann [LZ93] in the context of learning indexed families of recursive languages. They use the special property of their hypothesis space in which membership problem is decidable to generate negative examples.

[3] **COSINGLE** $= \{L \mid \text{card}(N - L) = 1\}$.

ACKNOWLEDGMENTS

We are grateful to Steffen Lange and Thomas Zeugmann for bringing this problem to our attention. John Case, Sanjay Jain, and Ross Quinlan provided helpful comments. We also thank the referees for suggestions. Research supported by Australian Research Council Grant A496004-56.

REFERENCES

[AS83] D. Angluin and C. Smith, A survey of inductive inference: Theory and methods, *Comput. Surveys* **15** (1983), 237–289.

[BB75] L. Blum and M. Blum, Toward a mathematical theory of inductive inference, *Inform. Control* **28** (1975), 125–155.

[CJLZ97] J. Case, S. Jain, S. Lange, and T. Zeugmann, "Incremental Concept Learning for Bounded Data Mining," Technical Report DOI-TR-136 Department of Informatics, Kyushu University, 1997.

[CS83] J. Case and C. Smith, Comparison of identification criteria for machine inductive inference, *Theoret. Comput. Sci.* **25** (1983), 193–220.

[Gol67] E. M. Gold, Language identification in the limit, *Inform. Control* **10** (1967), 447–474.

[HU79] J. Hopcroft and J. Ullman, "Introduction to Automata Theory, Languages, and Computation," Addison–Wesley, Reading, MA, 1979.

[LZ93] S. Lange and T. Zeugmann, Monotonic versus non-monotonic language learning, *in* "Proceedings of the Second International Workshop on Nonmonotonic and Inductive Logic," pp. 254–269, Lecture Notes in Artificial Intelligence, Vol. 659, Springer-Verlag, Berlin/New York, 1993.

[MF90] S. Muggleton and C. Feng, Efficient induction of logic programs, *in* "Proceedings of the First Conference on Algorithmic Learning Theory, Tokyo," pp. 368–381, Ohmsa, 1990. [Reprinted by Ohmsa Springer-Verlag]

[OSW86] D. Osherson, M. Stob, and S. Weinstein, "Systems That Learn, an Introduction to Learning Theory for Cognitive and Computer Scientists," MIT Press, Cambridge, MA, 1986.

[QUI90] J. Quinlan, Learning logical definitions from relations, *Mach. Learning* **5** (1990), 239–266.

[Sha81] E. Shapiro, "Inductive Inference of Theories from Facts," Technical Report 192, Computer Science Department Yale University, 1981.

[TB70] B. Trakhtenbrot and J. Bārzdiņš, "Konetschnyje Awtomaty (Powedenie i Sintez)," Nauka, Moscow, 1970. [in Russian]; English translation, "Finite Automata—Behavior and Synthesis," *Fundamental Studies in Computer Science* 1, North-Holland, Amsterdam, 1975.